Urbanity: A System for Interactive Exploration of Urban Dynamics from Streaming Human Sensing Data

Mengxiong Liu[1], Zhengchao Liu[1], Chao Zhang[1], Keyang Zhang[1], Quan Yuan[1], Tim Hanratty[2], and Jiawei Han[1]
[1]Dept. of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA
[2]U.S. Army Research Laboratory, Adelphi, MD, USA
[1]{mliu60, zliu80, czhang82, kzhang53, qyuan, hanj}@illinois.edu
[2]timothy.p.hanratty@mail.mil

ABSTRACT

With the urbanization process worldwide, modeling the dynamics of people's activities in urban environments has become a crucial socioeconomic task. We present `Urbanity`, a novel system that leverages geo-tagged social media streams for modeling urban dynamics. `Urbanity` automatically discovers the spatial and temporal hotspots where people's activities concentrate; and captures the cross-modal correlations among location, time, and text by jointly mapping different units into the same latent space. With `Urbanity`, the end users are able to use flexible query schemes to retrieve different resources (*e.g.*, POIs, hotspots, hours, activities) that meet their needs. Furthermore, `Urbanity` can handle continuous streams to update the learned model, thus revealing up-to-date patterns of urban activities.

1 INTRODUCTION

According to the World Bank statistics[1], more than half of the world's population lives in urban areas today. The enormous urbanization progress has made it a pressing need to systematically model people's activities in the urban space. While commercial location search engines (*e.g.*, Google Maps) allow users to retrieve relevant point-of-interests (POIs) with a bunch of keywords, they merely provide static knowledge about POIs without capturing the up-to-date dynamics underlying people's activities. Furthermore, their functionality is limited to retrieving POIs for keyword queries, and cannot handle flexible query schemes. For instance, a tourist in New York City may be interested in: what are the fun things to do at 9pm around my hotel? Instead of searching for POIs, such queries demand text-form activities for a given location (*i.e.*, my hotel location) and time slot (*i.e.*, 9pm), and cannot be readily answered by current location search engines.

The prevalence of geo-tagged social media (GTSM) sheds light on modeling urban dynamics in a data-driven way. By virtue of the popularity of GPS-equipped smartphones and various social media services, a tremendous amount of GTSM records have been created in the recent years. So far, more than 10 billion check-ins have been accumulated by Foursquare[2], and more than 10 million geo-tagged tweets are being created in the Twitterverse every day[3]. A GTSM record, which consists of a location, a timestamp, and a text message, is typically created by a user to share *on-the-spot* activities in real time. The massive amount of GTSM records can thus be viewed as millions of human sensors probing different areas in a city and reporting their activities online — which makes it possible to model urban dynamics in a data-driven manner.

We develop a system, named `Urbanity`, to model urban dynamics from massive and continuous GTSM streams and support interactive exploration of urban dynamics. `Urbanity` achieves scalable and effective urban activity modeling with a novel cross-modal representation learning method. It first automatically discovers the spatial and temporal hotspots representing the geographical regions and time periods in which people's activities concentrate. To support various query schemes, it then captures the cross-modal correlations among location, time, and text by jointly mapping different units into the same latent space. `Urbanity` also features an updating module that handles continuous GTSM streams to obtain up-to-date patterns of urban dynamics.

With a user-friendly Web interface, the `Urbanity` system is versatile and easy to use. The end users are able to use flexible query schemes to retrieve different resources (*e.g.*, POI, hotspots, hours, activities) that meet their information needs. As such, `Urbanity` is highly useful for both user-oriented and business-oriented scenarios. For example, a user can ask questions like "what are the popular locations to hang out with my friends at 9pm", or "what are the popular hours for hiking in a specified park". As `Urbanity` integrates GTSM streams with a POI database, the user can also retrieve fine-grained POIs (*e.g.*, a hotel or a pizzeria). As business-oriented examples, entrepreneurs can use `Urbanity` to examine the geographical hotspots of the target products when opening up new businesses, and a food truck owner can look for areas where people go to work and often choose fast food.

The contributions of `Urbanity` are: (1) a novel system that integrates continuous GTSM streams and POI database for modeling up-to-date urban dynamics; (2) a scalable cross-modal representation learning framework that supports online embedding of multiple modalities (location, time, text) from massive GTSM data; and (3) a versatile interface that allows end users to interactively retrieve

[1]http://data.worldbank.org/indicator/SP.URB.TOTL.IN.ZS (accessed May 11, 2017)

CIKM'17 , November 6–10, 2017, Singapore.

ACM ISBN 978-1-4503-4918-5/17/11…$15.00
https://doi.org/10.1145/3132847.3133177

[2]https://foursquare.com/about (accessed May 11, 2017)
[3]https://www.mapbox.com/blog/twitter-map-every-tweet/ (accessed May 11, 2017)

different resources (*e.g.*, region, hour, keywords, POI) with flexible query schemes.

2 THE URBANITY SYSTEM

2.1 Overview

Data. In Urbanity, we collect data for the Los Angeles city from two sources. The first is a POI database crawled from Foursquare[4], which consists of ~185 thousand POIs in Los Angeles. Each POI is described by a location, a name, a category (*e.g.*, "food", "travel & transport"), and a Web link. The second data source is the geo-tagged tweets published in LA. We monitor the Twitter Streaming API[5] and collect geo-tagged tweets with a bounding box. The crawler has been running since August 2014, and is able to collect more than 10 thousand geo-tagged tweets per day. Both the POI and geo-tagged tweet data are stored in a Mongo DB instance.

Architecture. Figure 1 shows the architecture of the Urbanity system. Generally, Urbanity consists of two major components: an urban dynamics modeling pipeline and a query processing component. The former consumes continuous GTSM streams and learns urban activity models in an online manner. It has three key modules: (1) a hotspot detector; (2) a cross-modal embedding learner; and (3) an online updating module. The latter leverages the learned model to process various queries from the end users and visualize the query results. At the core of it are a ranking module and a query optimizer. In what follows, we introduce the different modules in detail.

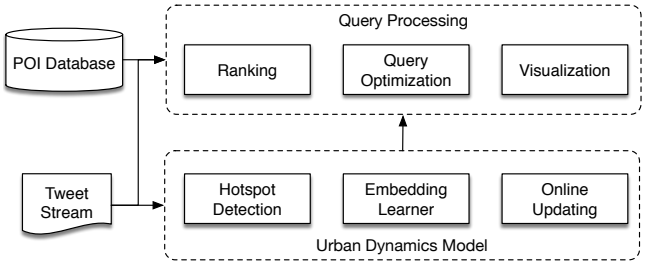
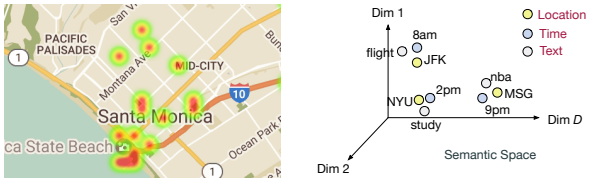


Figure 1: The architecture of the Urbanity system.

2.2 Urban Dynamics Modeling

2.2.1 Hotspot Detection. The hotspot detector is designed to automatically discover spatial and temporal hotspots from GTSM data. The spatial hotspots and temporal hotspots represent the geographical regions and time slots where people's activities in the city burst, as shown in Figure 2(a).

Formally, we define spatial and temporal hotspots as kernel density maxima in the two-dimensional and one-dimensional spaces, respectively. The hotspot detector is based on the mean shift algorithm [1], a non-parametric method for detecting kernel density maxima. However, the time complexity of the vanilla mean shift algorithm is $O(KN^2)$, where N is the number of data points, and K is the number of shifting steps. It is thus inefficient for million-scale data sets. To address this issue, we adopt an acceleration strategy [8], which partitions the space into equal-sized cells and pre-compute sufficient statistics (*e.g.*, number of points, sum of vectors) for each cell. During each shifting step, those precomputed statistics allow us to find the new kernel centers without accessing the points inside each cell. The complexity of our improved mean shift is $O(KN^2)$, which makes the hotspot detector scalable.



(a) Spatial hotspots. (b) Cross-modal embedding.

Figure 2: The hotspot detection and the cross-modal embedding modules.

2.2.2 Cross-Modal Representation Learning. The second key module in our urban activity modeling pipeline is the cross-modal embedding learner. It jointly maps different spatial hotspots, temporal hotspots, and textual keywords into the same latent space to preserve their mutual correlations. If two units are highly correlated (*e.g.*, the JFK airport area and the keyword "flight"), they tend to have similar representations in the latent space, as shown in Figure 2(b).

In Urbanity, the cross-modal embedding learner preserves two types of correlations during the learning process: (1) *the co-occurrence correlation:* two units (*e.g.*, the JFK airport area and the keyword "flight") that frequently co-occur in the same record are highly likely to be correlated and should have similar embeddings; and (2) *the neighborhood correlation:* two spatial (temporal) hotspots that are geographically (temporally) close to each other should have similar embeddings because of the space (time) continuity.

To realize the above intuition, we adopt a graph embedding procedure [8]. First, we construct a *correlation graph* to encode relationships between different units. The correlation graph consists of three different node types: (1) spatial hotspots; (2) temporal hotspots; and (3) textual keywords. And the graph edges are induced from either the co-occurrence correlation or the neighborhood correlation. We learn graph embeddings by minimizing the KL-divergence between the embedding-based neighborhood distribution and the ground-truth neighborhood distribution. Such an objective can be efficiently optimized based on stochastic gradient descent (SGD) and negative sampling [4].

2.2.3 Online Updating. People's activities in the urban space are continuously evolving rather than staying static. To reveal the most recent regularities of urban dynamics, Urbanity includes an online updating module that processes continuous GTSM streams. The updating module maintains a cache to keep newly arrived GTSM records and updates the model on a daily basis. To avoid overfitting newly arrived records, Urbanity regards the previously learned model as background knowledge and updates the model. Specifically, for hotspot detection, Urbanity considers old hotspots as *pseudo points*, where the weight of each point is set to the number

[4]https://developer.foursquare.com/
[5]https://dev.twitter.com/rest/public

of points attached to it. As new records arrive, Urbanity combines them with those pseudo points to detect the new hotspots efficiently. To update the cross-modal embeddings, Urbanity accommodates the new records by fine-tuning the previous embeddings. During the fine-turning process, we impose the constraint that the updated embeddings do not deviate much from the previous ones [9]. In this way, Urbanity generates embeddings that are optimized for new records while respecting the prior knowledge encoded in previous embeddings.

2.3 Query Processing

2.3.1 The ranking module. The Urbanity system allows end users to flexibly compose queries with any combinations of three primitive types: (1) location; (2) time; and (3) keywords. By including one or several of these primitive types, the users can retrieve useful information from Urbanity with various queries. Example queries include: What are the popular locations to hang out with my friends at 9pm? What are the popular hours for hiking in the Turkey Run State Park? For a user query, Urbanity retrieves top-k most similar items from the following categories: (1) spatial hotspot; (2) temporal hotspot; (3) keyword; and (4) POI. There can be multiple units in both a user query (*e.g.*, 9pm, 'friend') and a target item (*e.g.*, a POI is described by a location and several keywords). To compute the similarity score between a user query Q and a candidate C, we first derive the average embeddings for the units in Q and C, and then compute the cosine similarity, namely $\text{score}(Q, C) = \cos(\sum_{i \in Q} \mathbf{v}_i/|Q|, \sum_{j \in C} \mathbf{v}_j/|C|)$.

2.3.2 Query optimization. To answer a user query, it is time-consuming to go through all the candidate items in each category and find the top-k similar ones, because the number of candidates can be large in practice. We adopt an optimization strategy based on precomputation to handle user queries efficiently. Specifically, we pre-compute and cache the top-N ($N > k$) neighbors, in terms of cosine similarity, for each unit. At query time, we apply the Threshold Algorithm [2] to aggregate the pre-computed neighbors to obtain the top-k result list. With $N = 100$, such an optimization enables our system to process a query with multiple keys in less than 100 milliseconds.

3 DEMONSTRATION PLAN

3.1 The Web Interace

As shown in Figure 3, the Web interface of Urbanity consists of two panels. The function of the left panel is two-fold: (1) specifying the textual and temporal query primitives; and (2) displaying top similar keywords and time slots. The right panel is designed for specifying query locations, as well as displaying spatial hotspots and POI results. At query time, users can compose their queries using one or several of the following primitive operations: (1) specifying a set of keywords using the text input box; (2) specifying a time slot using a sliding bar; and (3) specifying a set of locations by clicking on the map. Furthermore, users can provide other conditions in their queries, such as using the POI toggle to determine whether POIs should be displayed on the result map, and using a timeline handler to visualize the temporally evolving activities for any location throughout the day.

After 'submit' is clicked, Urbanity retrieves the results and display the top-k most relevant time slots and keywords at the bottom of the left panel. Meanwhile, it visualizes the top-k spatial results on the map in two different ways: (1) a heatmap for visualizing the top-k spatial hotspots for the query; and (2) the top-k relevant POIs pinned on the map. A user can click any POI to view its detailed information, including category, name, website URL, *etc.*.

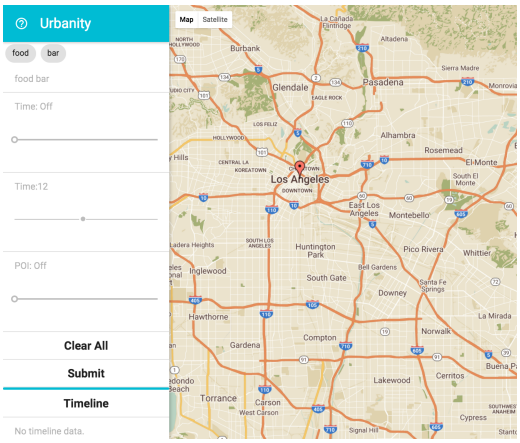


Figure 3: The Web interface of Urbanity.

3.2 Illustrative Query Schemes

As aforementioned, Urbanity supports multiple query schemes for interpreting urban dynamics. In our demonstration, we will showcase the usages of different schemes and how to interpret the results. We describe several examples in the following.

Textual queries. Figure 4 shows the results for a textual query, which is composed with two keywords: "beach" and "swim". As shown, the locations are mostly located in famous beach areas in Los Angeles such as Santa Monica and Long Beach, and the red markers in the map denote the popular POIs. For instance, one result POI is "Smackfest", an organization in Redondo Beach that hosts beach volleyball events.

Spatial queries. Figure 5 shows the results after a user selects locations close to the LA international airport. Again, one can see

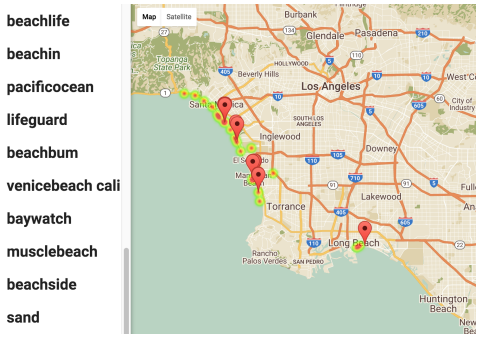


Figure 4: Results for the textual query 'swim + beach'.

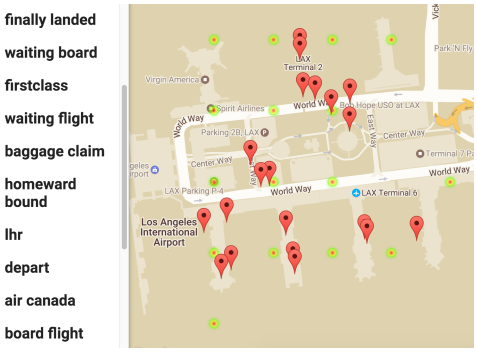


Figure 5: Results for the spatial query at the LAX airport.

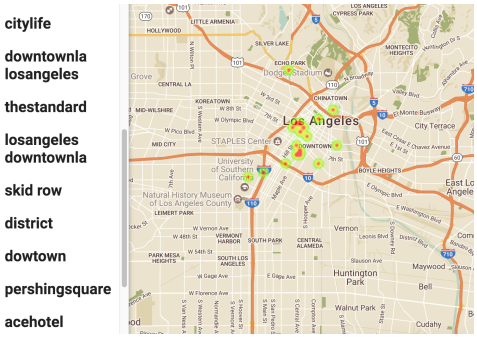


Figure 6: Results for the query 'downtown + 10am'.

the resulting words are closely related to flight-related activities ('depart', 'Air Canada', *etc.*).

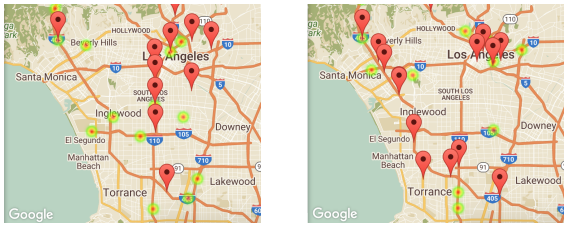
Temporal-textual queries. Besides specifying one primitive type in their queries, users are also allowed to combine different types to retrieve the information they need. For example, Figure 6 shows the results for a temporal-textual query, where the user specify a keyword 'downtown' and a time slot '10 am'. The results can accordingly reflect people's activity patterns in the morning in downtown LA.

3.3 Visualizing Activity Evolutions

Another useful feature of Urbanity we plan to demonstrate is its ability to visualize the temporal evolutions of people's activities. Take the temporal-textual query as an example. Assume the query keywords are fixed, the user can drag the sliding bar on the left pane of Urbanity to visualize the dynamic evolvements of people's activities throughout the day. Figure 7 shows an example of the query 'traffic'. We can see that Interstate 110 (middle of LA) is busy around noon, while Interstate 405 (left of LA) is busy around 6pm. Such evolving visualizations can provide intuitive interpretations about the dynamic urban activities.

4 CONCLUSION AND FUTURE WORK

We have described Urbanity, a system for exploring urban dynamics using massive human sensed social media. It comes with



(a) Results at 11am. (b) Results at 6pm.

Figure 7: The temporal evolution for the query 'traffic'

a non-parametric spatiotemporal hotspot detector, a novel cross-modal embedding method, a bunch of optimization strategies, and finally an easy-to-use Web interface. We believe Urbanity will be useful for various urban computing and planning applications. In the future, we plan to further develop Urbanity in three different ways. First, it is interesting to enhance Urbanity with recent techniques for detecting emergent local events from the geo-tagged social media streams [7, 10]. Second, we intend to develop visualization modules that analyze the spatiotemporal dynamics of different terms [3, 5]. Finally, it is promising to integrate human sensing data with conventional sensor data [6] for more comprehensive urban activity modeling.

Acknowledgements: This work was sponsored by the U.S. Army Research Lab. under Cooperative Agreement No. W911NF-09-2-0053 (NSCTA), National Science Foundation IIS-1017362, IIS-1320617, and IIS-1354329, HDTRA1-10-1-0120, and Grant 1U54GM114838 awarded by NIGMS through funds provided by the trans-NIH Big Data to Knowledge (BD2K) initiative (www.bd2k.nih.gov), and MIAS, a DHS-IDS Center for Multimodal Information Access and Synthesis at UIUC. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing any funding agencies.

REFERENCES

[1] Dorin Comaniciu, Peter Meer, and Senior Member. 2002. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24 (2002), 603–619.
[2] Ronald Fagin, Amnon Lotem, and Moni Naor. 2003. Optimal aggregation algorithms for middleware. *J. Comput. Syst. Sci.* 66, 4 (2003), 614–656.
[3] Theodoros Lappas, Marcos R. Vieira, Dimitrios Gunopulos, and Vassilis J. Tsotras. 2012. On The Spatiotemporal Burstiness of Terms. *PVLDB* 5, 9 (2012), 836–847.
[4] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*. 3111–3119.
[5] Jinfeng Rao, Xing Niu, and Jimmy J. Lin. 2016. Compressing and Decoding Term Statistics Time Series. In *ECIR*. 675–681.
[6] Shuochao Yao, Shaohan Hu, Yiran Zhao, Aston Zhang, and Tarek F. Abdelzaher. 2017. DeepSense: A Unified Deep Learning Framework for Time-Series Mobile Sensing Data Processing. In *WWW*. 351–360.
[7] Chao Zhang, Liyuan Liu, Dongming Lei, Quan Yuan, Honglei Zhuang, Tim Hanratty, and Jiawei Han. 2017. TrioVecEvent: Embedding-Based Online Local Event Detection in Geo-Tagged Tweet Streams. In *KDD*. 595–604.
[8] Chao Zhang, Keyang Zhang, Quan Yuan, Haoruo Peng, Yu Zheng, Tim Hanratty, Shaowen Wang, and Jiawei Han. 2017. Regions, Periods, Activities: Uncovering Urban Dynamics via Cross-Modal Representation Learning. In *WWW*. 361–370.
[9] Chao Zhang, Keyang Zhang, Quan Yuan, Fangbo Tao, Luming Zhang, Tim Hanratty, and Jiawei Han. 2017. ReAct: Online Multimodal Embedding for Recency-Aware Spatiotemporal Activity Modeling. In *SIGIR*. 245–254.
[10] Chao Zhang, Guangyu Zhou, Quan Yuan, Honglei Zhuang, Yu Zheng, Lance Kaplan, Shaowen Wang, and Jiawei Han. 2016. GeoBurst: Real-Time Local Event Detection in Geo-Tagged Tweet Streams. In *SIGIR*. 513–522.